The DoD Software Tech News

April 2002

Vol. 5, No. 2

Software Engineering Education

http://iac.dtic.mil/dacs/

About the DoD Software Tech News (STN)

Article Reproduction

Images and information presented in these articles may be reproduced as long as the following message is noted:

"This article was originally printed in the *DoD Software Tech News*, Vol. 5, No. 2. Requests for copies of the referenced newsletter may be submitted to the following address:

Lon R. Dean, Editor
Data & Analysis Center for Software
P.O. Box 1400
Rome, NY 13442-1400

Phone: 800-214-7921
Fax: 315-334-4964
E-mail: news-editor@dacs.dtic.mil

An archive of past newsletters is available at www.dacs.dtic.mil/awareness/newsletters/."

In addition to this print message, we ask that you <u>send us three copies</u> of any document that references any article appearing in the *DoD Software Tech News*.

About This Publication:

The *DoD Software Tech News* is published quarterly by the Data & Analysis Center for Software (DACS). The DACS is a DoD sponsored Information Analysis Center (IAC), administratively managed by the Defense Technical Information Center (DTIC) under the Defense Information Systems Agency (DISA). The DACS is technically managed by Air Force Research Laboratory, Rome, NY and operated by ITT Industries, Advanced Engineering and Sciences Division.

DACS
P.O. Box 1400
Rome, NY 13442-1400

Phone: 800-214-7921
Fax: 315-334-4964
E-mail: dacs@dtic.mil
URL: http://iac.dtic.mil/dacs

Data & Analysis Center for Software
http://iac.dtic.mil/dacs

Distribution Statement:

unclassified and unlimited

To Subscribe to this Publication Contact:

Phone: 800-214-7921
Fax: 315-334-4964

E-mail: news-editor@dacs.dtic.mil
Web: www.dacs.dtic.mil/forms/regform.shtml

Tech Views

by Morton Hirschberg, STN Editorial Board Chairman

This issue inaugurates Tech Views, an editorial column focusing on the articles featured here. Tech Views will provide commentaries and/or augmentation regarding the theme of each issue. One or more members of the Editorial Board will contribute to each Tech Views editorial.

The articles appearing in this issue have an underlying theme - 'Software in Crisis'. A crisis of product rather than process and a crisis in education. Several programs dealing with these issues are presented.

Computer Science and Software Engineering Education is vital for the continued success of communications and weapons development for the DoD. The Association for Computing Machinery (ACM) and the Computer Society of the Institute for Electrical and Electronics Engineers (IEEE-CS) have jointly developed the Curricular Guidelines for Undergraduate Programs in Computing (see http://www.acm.org/sigcse/cc2001). This important document provides guidance for the matriculation of entry level Computer Scientists and Software Engineers.

To continue their education from entry-level positions to fully skilled journeymen practitioners and beyond requires both on-the-job training, which provides realism, and additional education. The latter can range from a course or two to advanced degrees at the Masters and Ph.D. levels. Such instruction may be obtained through the classroom and now through distance learning via hard documentation (texts, videotapes, etc.) or the Internet.

Many institutions now offer distance learning. A prime example is given in the Baker and Michel paper, which examines software acquisition. Further examples of distance learning may be found at http://www.ed.gov, which details the Distance Education Demonstration Program, listing participants and programs.

The paper by Jack Jensen details Masters and Ph.D. level programs available from the Naval Post Graduate School in Monterey, CA. Courses may be taken via distance learning. More importantly, these programs are designed with civilian employees of the DoD in mind. Course materials use DoD examples and thesis work can be related directly to the student's own job. Courses taken which lead to a certificate of training may be applied toward obtaining an advanced degree at a later date.

Bernstein and Klappholz take the approach of teaching using real production cases and 'working in the large'. This as opposed to small world problems, which are typically small and process oriented.

Education, Information Technology, and the ‘Software Crisis’

Lawrence I. Baker, Defense Acquisition University and
L. John Michel III, National Defense University

Ever since the “software dark ages”, prior to circa 1975, and even during, many people have called attention to the so-called ‘software crisis’ and how it is causing all manner of problems in our high technology society. It is our belief that this focus, no matter how well intentioned, has been misplaced at best, and at worst is completely misleading. The concentration should be on the product, not the process. It may very well be, as the Total Quality Management (TQM) advocates claim, that the quality of the process determines the quality of the product; but unless one has a clear definition and expectation for the product, the process is irrelevant. It is analogous to the Chinese belief that states (paraphrased), “If you don’t know where you want to go, a map is useless”; and, accordingly, “Any road will do.” In other words, we believe that Dr. W. Edwards Deming’s reluctance to embrace TQM was correct; and his ideas relative to ‘profound knowledge’ being the key to enterprise, organization, or system success are being overlooked. Basically, he was looking for answers to the following. What is it that we want to accomplish? What are the useful products of the endeavor? How will it benefit the stakeholders? The “accepted” means to successfully accomplish the objectives are controversial, ranging across a spectrum that is bounded by the Software Engineering Institute’s Capability Maturity Model at one end to Extreme Programming at the other, with a 6-sigma compromise approach somewhere in between. Each has its proponents, merits, and demerits; however, it is clearly beyond the scope of this paper to discuss it beyond stating that each can be shown to be more or less advantageous depending on the situation.

Thus, the critical area of focus should be the outcomes; the means of getting there are myriad. We assert that there is definitely a crisis and the unpredictable, ‘software dragon’ is part of it; but there is a larger issue that must be addressed from the system, organization, or the enterprise point of view. Ultimately, if one looks at the situation from that perspective, it is the information that becomes paramount. Thus, it is the *management of the information and the technology that supports it* which should be of utmost concern to us; and it applies regardless of the domain. Whether for a corporate enterprise, a government agency, a national security system, an intelligence system, a weapons systems, a C^4I system, a management information system, or some other application, unless the human as an individual, or the group, or society in general is

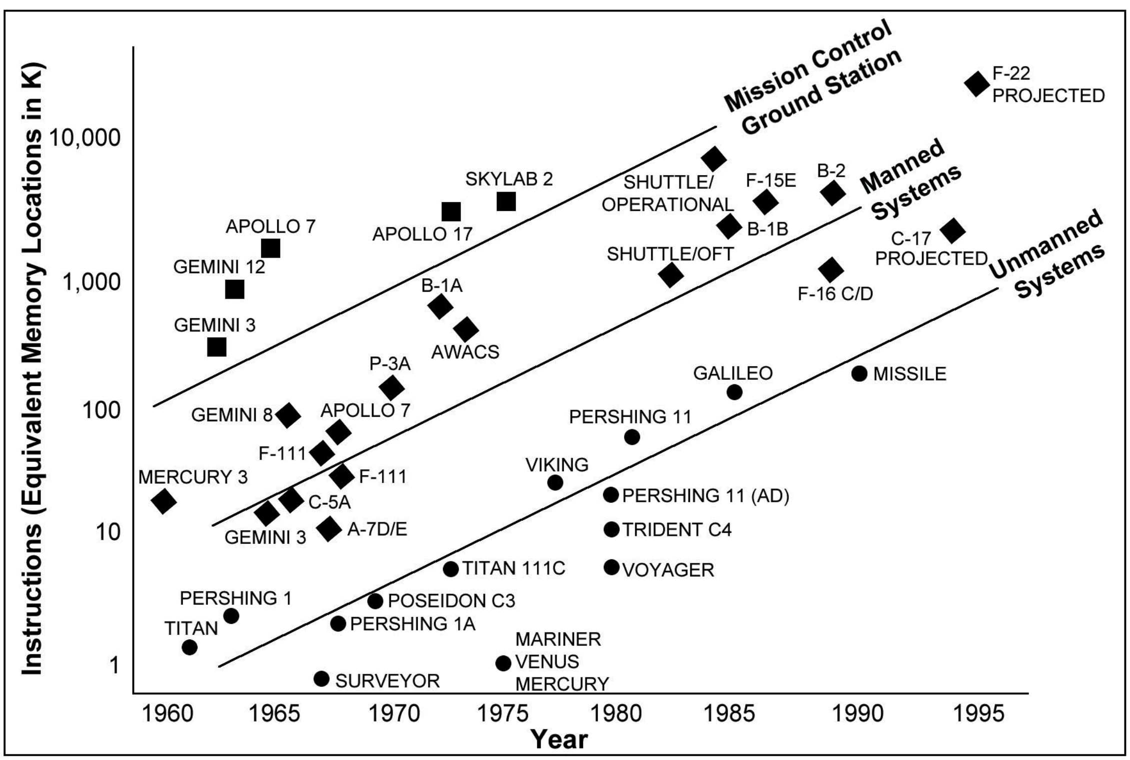
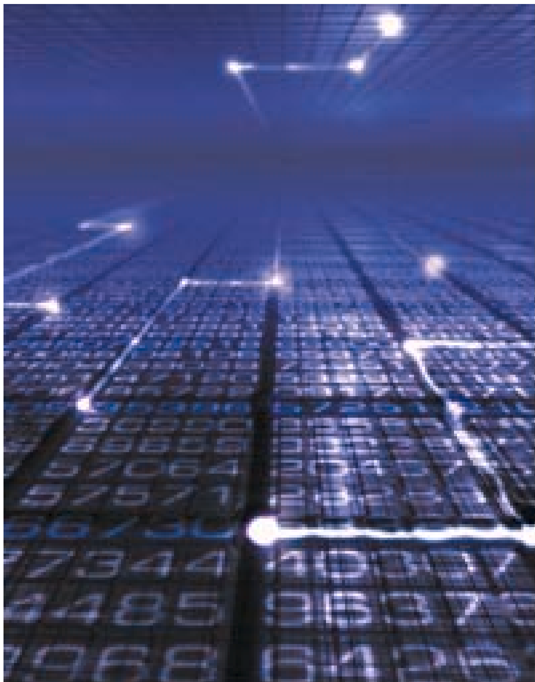


Figure 1. Software Trends



economically provided what he, she, or they require in a timely manner, all else can be considered to be of little matter.

Given that, let's look at where we are. Our systems are increasingly becoming more software intensive because software is replacing the functionality formerly performed by people and hardware. Rear Admiral Robert M. Moore, former commander of the Naval Information Systems Management Center, identified this transition in March 1993, when he stated, "At one time, it was the hardware that supported the mission. Today, the hardware is rather generic, capable of supporting any mission. It is the software that provides the real functionality." Figure 1 illustrates this point; although presenting information only through 1995, the trend has continued.

In a 1992 report to the House Armed Services Committee, the General Accounting Office estimated that total annual software cost would account for twenty percent of the Department of Defense's (DoD) budget by 2008 [1]. In October, 1997, Federal Sources, Inc. completed a survey of defense spending on software used for weapons systems, information systems, and command, control, communications, computers, and intelligence systems [2], excluding software for non-tactical systems. The report projected that, in 2002, DoD will spend over $20 billion annually on such software.

Software is the critical component of today's defense systems. Yet, software acquisition and development within DoD continues to be a significant management problem. A variety of studies and analyses over the past twenty years have continued to identify significant systemic software acquisition problems and an unabating need for software knowledgeable acquisition management professionals. In September, 1987, the Defense Science Board (DSB) Report on Military Software, Office of the Under Secretary of Defense for Acquisition, stated, "Application-knowledgeable, technically skilled leaders are the military's *limiting resource* in acquiring today's computer technology. … Few program offices are staffed [properly] due to a shortage of qualified people. ... the DoD should implement the education and training necessary for its *software acquisition management personnel* to master both software technology and *acquisition management*." We would further amplify this statement by adding that the educational process must emphasize that the technology (software) supports the management of the information.

The DoD Software Master Plan, Vol. I, February, 1990, developed by the Defense Acquisition Board Science and Technology Committee, reported, "Improving software education and training is critical. ... there is a need to coordinate efforts of the National Defense University (NDU), the Defense Systems Management College (DSMC), and the Industrial College of the Armed Forces to integrate software *acquisition* and development programs into existing courses and to establish mandatory software engineering education for all DoD technical and contractual personnel involved in the *acquisition* process."

The DoD Information Systems (IS) Work Force Education, Training and Career Development Executive Resources Task Force Report (October, 1992) asserted, "Technical vitality of the IS work force is critical to effectively deploy information systems in support of the DoD war-fighting mission. The need to provide recurring technical training to individuals, *especially at mid-*

Continued on page 6

career and executive levels, was communicated throughout our meeting with services, agencies, and private industry. This training is essential to ... keep pace with the acquisition of more advanced computer and telecommunications systems."

In November, 2000, the DSB Task Force on Defense Software, Office of the Under Secretary of Defense for Acquisition and Technology, stated unequivocally that, "If we are to establish an elite information force, the following actions should be taken: 1) Institute mandatory software-intensive systems training for program managers and key staff on all ACAT programs; 2) Require collaborative government/ contractor team training at program start and at critical milestones; 3) Require annual software technology refresher training for joint government/contractor teams; 4) Develop a graduate-level program for software systems development and acquisition training; and 5) Require that each ACAT program office appoint an expert software systems architect."

"Technology is dominated by two types of people: those who understand what they do not manage and those who manage what they do not understand" – Anonymous. Without question, there is a critical need for a work force highly trained in the complex programmatic discipline of software acquisition management. It continues to be a daunting task to educate members of the DoD acquisition work force who acquire, develop, engineer, test and evaluate, conduct research on, or procure software-intensive systems. This has been amplified in the past by a dearth of appropriate educational opportunities.

Aristotle stated that, "The educated differ from the uneducated as much as the living from the dead." The need for a review of the DoD's software acquisition management education and training curricula and career programs was reinforced in May, 1993 by the Acquisition Management Functional Board, an organization that advises DoD component executives in the management of accession training and career development of acquisition work force personnel. On Oct. 19, 1993, the 'terms of reference' for the review of software acquisition management education was approved by Ms. Colleen A. Preston, then Deputy Under Secretary of Defense for Acquisition Reform.

Approved in March, 1994, the review team's report established a set of nine critical competencies and twenty-four key competency areas. It is important to note that the competencies address and encompass all of the relevant and pertinent disciplines, not just software. It is insufficient for an acquirer to have a "stovepipe" view of software; the "bigger picture" must not be ignored. In addition, the report contained the recommendation for the development of assignment-specific mandatory courses for software acquisition personnel for Level I, II, and III training career levels. Level I courses would be informational in nature; students would be provided the basis for understanding. Level II would be knowledge-based and practitioner oriented. Level III would be knowledge management oriented with emphasis on the decision making process. Under the auspices of the Defense Acquisition University, the Information Resources Management College (IRMC), located at NDU, and DSMC began joint development of an evolutionary course curriculum. IRMC was tasked to lead the design of the courses for the communications and computer career field and the Level III software capstone courses; DSMC led the design of the Level I basic and Level II intermediate software courses.

The developed courses are Level III – IRM303, SAM301; Level II – IRM201, SAM201; and Level I (distance learning via the Internet) – IRM101, SAM101. The courses are essentially equivalent; the differences tend to reflect the special needs of the community that each institution generally serves – IRMC for the ASD(C^3I) constituency and DSMC for the USD (AT&L) constituency. The developers of the courses did achieve the goal of incorporating the theme that successful outcomes of the acquisition process require that the technology supports the management of information. The following table shows the specifics for each of the courses.

Since there is no true software

Table 1. Mandatory Courses for Software Acquisition Personnel

Course	Target Audience	Format	Goal
IRM 101	GS-9 and below and military ranks 01-03	Distance education, 19 lesson modules.	Exploration of introductory level concepts involved in DoD information systems acquisition management.
SAM101	GS-9 and below and military ranks 01-03	Distance education, 18 lesson modules.	Exploration of key aspects of software acquisition management.
IRM 201	Level I certified mid-level managers with responsibilities in information systems/ information technology acquisitions.	10-day classroom -based curricula.	Development of competence in applying IS/IT management skills in IS/IT planning, organizing, directing, and controlling information systems acquisition programs.
SAM 201	Experienced, intermediate-level acquisition personnel who fill or are slated to fill software acquisition management positions.	10-day classroom -based curricula.	Development of competence in acquisition management of software-intensive weapon systems, command and control systems, and automated information systems.
IRM 303	Senior-level managers with responsibilities in information systems/ information technology acquisitions.	14-day classroom -based curricula.	Mastery of skills in evaluating and recommending strategies, evaluating plans, and making decisions in IS/IT acquisition management by using current technology to perform authentic tasks in a realistic, team oriented environment.
SAM 301	Senior-level managers with responsibility for programs in which software is a critical component.	10-day classroom -based curricula.	Mastery of key software acquisition competencies established for Level III acquisition professionals, while concentrating on software specific considerations.

career field at the present nor does there appear to be any impending groundswell for one, the software acquisition management courses remain assignment-specific, providing unique opportunities to acquire the knowledge required for a specific job or position. However, they have also been identified by the Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics as integral to the education of all acquisition work force personnel. They allow the individuals to maintain proficiency by remaining current with legislation, regulation, policy, and current practice. The courses are for people who acquire,

Continued on page 8

develop, engineer, test, evaluate, conduct research on, and procure software-intensive systems.

Carl Sandburg has said, "Sometime they'll give a war and nobody will come." Few 'dragon slayers' have taken up their swords and joined the crusade to slaughter the dragon. Given the criticality of the situation, the numbers of attendees have, in the past, been abysmal. Why the disappointing response? Perhaps, it is because software acquisition is accurately recognized as not being a career field of its own; perhaps, it is because the realization has not sunk in that (almost) all systems are information systems and the ubiquitous 'thing' that collects, transfers, manipulates, presents, secures, assures, and archives that information is software; perhaps it is the draw down of the acquisition work force; or perhaps it is because software acquisition courses are perceived as just another set of educational requirements that takes the participant out of his or her workplace.

For whatever reason, the "software education crisis" is not being rectified, and the 'dragon' still roams the land. The students' evaluations of the courses indicate that the colleges have developed quality programs that meet the needs of the software acquisition professional. This shortfall means that seats are readily available. In the context of the overwhelming individual and societal needs, the excuses are meaningless and unavailing. Malcolm Forbes has been quoted as saying,

> *"Education's purpose is to replace an empty mind with an open one."*

Those who agree should visit one of the following web sites or contact the authors of this article.

http://www.ndu.edu
http://www.dau.mil

The views in this article are those of the authors and do not reflect the official policy or position of the Defense Acquisition University, the National Defense University, the Department of Defense, or the U.S. Government.

Author Contact Information

Mr. L. John Michel, III
300 5th Avenue
Marshall Hall (Building 62)
Fort Lesley J. McNair, DC 20319

Phone: 202-685-2062
DSN: 325-2062
E-mail: Michell@ndu.edu

Mr. Lawrence I. Baker
Defense Acquisition University
9820 Belvoir Road
Fort Belvoir VA 22060

Phone: 703-805-3636
DSN: 655-3636
E-mail: larry.bakerl@dau.mil

About the Authors

Lawrence I. Baker is a Professor of Engineering Management in the Technology and Engineering Department and the Program Director for information technology course development at the Defense Acquisition University. He was formerly a member of the Software Acquisition Management Department at the Defense Systems Management College. Including his last nine years of government service, his background includes over 40 years of systems, software, and hardware experience in information technology with an emphasis on collection, transmission, processing, presentation, securing, and archiving. Mr. Baker earned his Baccalaureate Degree in Electrical Engineering (BEE) from the City College of New York in 1959 and his Master of Science (Systems Management) from the University of Southern California in 1986.

L. John Michel is a Professor of Systems Management at the Information Resources Management College, the National Defense University. Currently, he manages the course: Assuring the Information Infrastructure. He formerly managed Advanced Software Acquisition Management and Software Management for Executives. He has over 22 years experience in programmatic aspects of software intensive systems in the command and control, intelligence, and personnel communities. He has a BBA and MBA from the University of Georgia.

References

1. "Guidelines for Successful Acquisition and Management of Software-Intensive Systems", Air Force Software Technology Support Center, June 1996.

2. General Accounting Office Report, IMTEC-92-62BR.

3. Seffers, George I., "Survey: Software Cost to Exceed Hardware," *Defense News,* Nov. 10-16, 1997, p. 6

Software Engineering Graduate Opportunities at the Naval Postgraduate School

by Capt. Jack Jensen, Software Eng. Automation Center
U.S. Naval Postgraduate School

Today the cost of software could easily account for twenty percent of the Department of Defense budget according to Paul Strassman, a former Director of Defense Information. That would equate to an annual expenditure of over $60 billion, a huge investment by any standard. Any marginal improvement in the effectiveness of this software or the efficiency of its acquisition would yield an enormous return on the investment in both the near term, and over the life cycle of the systems being acquired.

DoD software systems are typically highly complex and may also be real-time and mission critical, so any cost savings gained are unlikely to come from simple, easy actions or "quick fixes." Realizing this, the Navy concluded that well-educated managers would have the greatest impact on effectiveness. Decision makers who understand the software acquisition process and have an in-depth knowledge of the principles of good software engineering represent the most cost-effective way to improve the overall quality and reduce the life cycle cost of tomorrow's software systems. Topics of concern include software architecture, system requirements, software design, challenges, and risks.

In support of this concept, a unique and highly innovative program in graduate education in software engineering has been undertaken at the U.S. Naval Postgraduate School, Monterey, California. The program offers both Masters and PHD degrees in software engineering. Although some students are in residence on the campus in Monterey, most are gaining their degrees through Distance Learning. These classes are conducted via video teleconferencing primarily, and involve both military and DoD civilians from activities across the country, and a few stationed aboard ships or overseas.

The curriculum covers all the principles of superior software engineering and the software engineering body of knowledge, but it's tailored to be particularly relevant to the current and emerging needs of the Navy and DoD. Thus, topics uniquely applicable to DoD systems are the prime focus of this program. Typically, these receive little emphasis at other universities. One such focused course is the guaranteed performance and safety of software systems – a topic of great importance for weapons systems, missile systems, and others. Other courses cover the development of embedded real time systems; software engineering research and development in DoD; engineering network centric systems; and automated software/hardware integration in DoD; among other uniquely relevant topics to DoD activities.

Table 1. Part-Time MSSE Course Matrix

1 - Fall	SW3460 (3-1) Software Methodology	SW4582 (3-1) Software Safety
2 - Winter	SW4591 (3-1) Requirements Engineering	SW4580 (3-0) Design of Embedded Real-Time Systems
3 - Spring	SW4590 (3-1) Software Architecture	IS4300 (3-2) Software Engineering & Management
4 - Summer	SW4540 (3-1) Software Testing	MN3309 (4-1) Acquisition of Embedded Weapon Systems Software
5 - Fall	SW4592 (3-1) Software Risk Assessment	IS3301 (3-2) Fundamentals of Decision Support Systems
6 - Winter	SW4500 (3-1) Software Engineering	EO4011 (3-2) Systems Engineering for Acquisition Managers
7 - Spring	SW0810 (0-8) Thesis Research	SW0810 (0-8) Thesis Research
8 - Summer	SW0810 (0-8) Thesis Research	SW0810 (0-8) Thesis Research

Table 1 – Curriculum schedule for a civilian to obtain a Master of Science degree in Software Engineering in two years via distance learning.

Students in the Masters Program are typically mid-level DoD managers and engineers, and are frequently working within development commands for major DoD systems. The program is normally two years but some students may accelerate or slow their pace for personal or professional reasons. Regular class sessions are conducted each week and students "attend" via video teleconference. The MS degree requires twelve graduate level courses and a thesis. A distance learning course matrix is shown in Table 1.

For those students who are particularly gifted and who wish to become involved in more research associated with software engineering, the PHD degree program is also available.

The Naval Postgraduate School was the first university to offer a PHD in Software Engineering, and it remains one of the very few opportunities to earn this degree via distance learning. Again, students are typically military or civilian professionals in mid-career and will work on the PHD degree over a period of approximately three years.

Additionally, for students or organizations not fully ready to commit to a full graduate degree, short-term programs can be tailored to lead to Software Engineering

Continued on page 12

Certificates in particular areas. These courses are taught at the graduate level. If the student takes a real interest, and later decides to pursue a graduate degree, a number of graduate credits will already be available to apply to the overall credit requirement needed for satisfactory completion.

Unique to this graduate-level distance learning program is the focus of the research associated with the thesis or dissertation. It is typical for students to be attracted to the program because they are particularly interested in improving some aspect of their job or the process associated with the software systems that they are acquiring. Thus, they choose research topics which are relevant to their own situation and their own commands.

Thus, in addition to potentially resolving the original technical problem, their graduate education leads to long term improvement in better and more capable software system managers, better systems, and greater effectiveness within their organizations. And all the time that the student is getting the graduate degree, he or she is still on the job - just doing it better each month. From the employer's perspective, there is also a big advantage to retaining educated and top performing individuals within the command or organization. Those returns on any investment are hard to beat!

Finally, the cost of each program is very competitive with the cost of getting graduate degrees from other universities. The costs can be underwritten by a variety of methods from total underwriting by the parent organization; to research support for specific issues/tasks; to fellowships; to various tuition assistance programs.

The center of software engineering research activity associated with the educational program at NPS is the Software Engineering Automation Center. It is positioned to afford maximum flexibility in achieving high quality research in software engineering while facilitating excellence in software engineering education. The web site for additional information is http://seac.nps.navy.mil.

Author Contact Information

Jack Jensen, CAPT USN (Ret)
Software Eng. Automation Center
U.S. Naval Postgraduate School,
833 Dyer Rd.
Monterey, CA 93943

Phone: (831) 656-2168
E-Mail: jjensen@cs.nps.navy.mil
URL: www.nps.navy.mil

About the Author

Jack Jensen, is the Managing Director of the Software Engineering Automation Center, at the U.S. Naval Postgraduate School.

CAPT Jensen has broad experience in the development and acquisition of DoD software and in computer systems management. His Navy experience included tours on the SECNAV staff, as Program Manager for Navy Environmental Systems, and as the Commanding Officer of Fleet Numerical Meteorology and Oceanography Center, one of DoD's largest supercomputer centers. His civilian experience includes managing the development of software systems and the education and training of computer systems professionals.

References

1. Federal Computing Week, April 26, 1999, http://www.fcw.com/fcw/articles/1999/FCW_042699_385.asp

SEAC

Innovative Software Engineering Education

By Professors Lawrence Bernstein and David Klappholz, Stevens Institute of Technology, Hoboken, NJ

Software projects often fail because the staff lacks Software Engineering education or when they had it they fought it. To compound the problem they don't accept Software Best Practices. Our challenge is to overcome the natural biases of software professionals and computer science students. Reading case histories of failed software tends to convince some students of others' stupidity. While other students intellectually accept the existence of the problems, just reading about them does not convert many at the 'gut level.' At the gut level one sits up, takes notice, and does something different.

Motivation

Our approach is to force students to live through specific case histories, each one chosen to get across a small number of important issues. The method works. Students internalize the software engineering lessons and follow best practices to avoid the traps they experienced.

Here is how the approach works. First, a set of software process issues is selected. Here are the ones we chose for our first live-through case history:

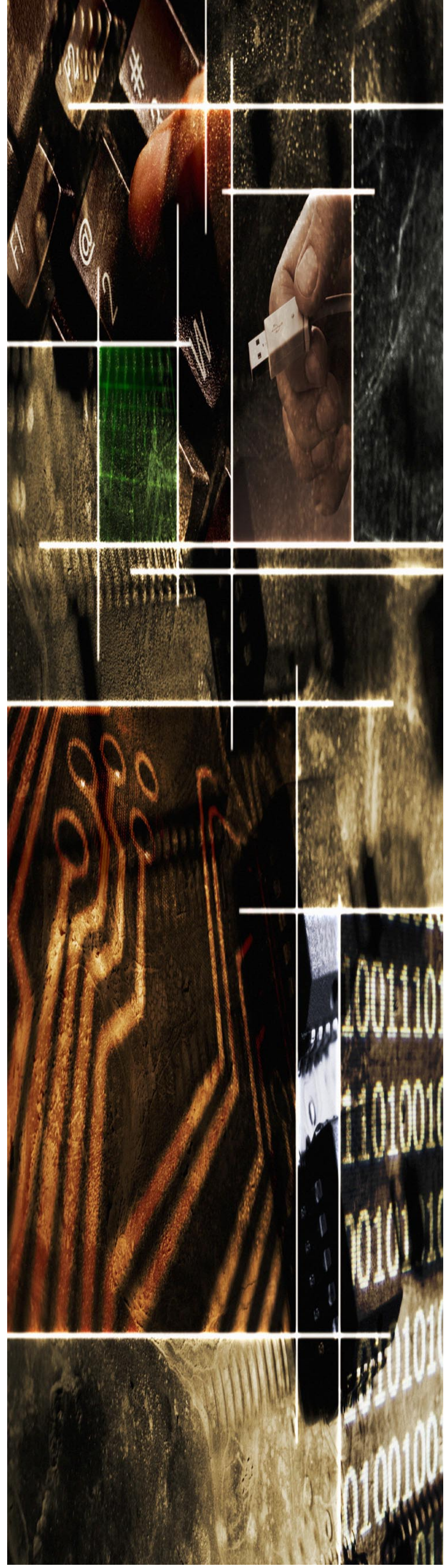
1. the need to have close customer/user relations
2. the need for up-to-date documentation throughout the life of the project
3. the need to identify risks and to develop contingency plans
4. the need to account for human foibles

Second, a case history based on a project facing these challenges is chosen. Students are not given the entire case history up front; rather, they are given the same problem/project as the actual developers who executed the case history faced. They are given no more information about the problem/project than were those developers. The project information is simplified to ease understanding.

Background

Computer Science is the study of the technology (State-of-the-Art) involved in the development of computer software. As it is usually taught in a post-secondary setting, Computer Science deals with "programming in the small," i.e., one-person or few-person software projects. Software Engineering, on the other hand, is the study of the method or process (State-of-the-Practice) whereby production software is developed - "programming in the large." State-of-the-Practice includes both engineering practices and project management or group dynamic processes. Many post-secondary programs in Computer Science offer a Software Engineering or Senior Project course as a capstone.

Continued on page 14



Because of the very different natures of technology on the one hand and method/process on the other, and because computer science students are typically technology-oriented and method/process-averse, the typical Software Engineering course reaches far fewer future software developers than suits the best interests of either the students themselves or the software industry at large. We have developed a novel instructional method, the Live-Through Case History method for addressing this problem, have developed a first live-through case history, and have used it successfully in the first few weeks of a two-semester undergraduate Software Engineering course.

The result was that students were shocked into an awareness of the issues and how to deal with them in six weeks of two classes a week. One class meeting was devoted to project meetings and the other to lectures on Software Engineering topics including other case histories.

Conducting the Case History

Because there would be only one live-through case history in our Senior Project course, we had to choose one that would achieve the greatest effect in the limited time available. We chose the case history of a brief development project that one of the authors worked on, in 1985, as a public service project. The problem/project was that of automating an elementary school library's manual system for generating overdue-book notices.

The class of forty students was divided randomly into four equal-size development teams. Students were given the same details, as were the original software developers in the case history. The instructor played the role of the customer, the school librarian, and was available to students, to respond to questions, both in class and by e-mail. Students were told that the customer would evaluate their work, exactly as work is evaluated in the real world.

Results

As is frequently the case in real software development projects, the overdue book notice project had a hidden requirement that was so obvious to the customer that she failed to mention it; it is that overdue notices must be sorted, first by teacher name, then, for each teacher by class, and, finally, within each class by student's family name. The system analyst rejected the real software system when she first saw it. The original developers failed to elicit the hidden make-or-break requirement, and thus failed to satisfy it. Each of the student teams fell into this same trap and they learned the lesson of the need to find any 'hidden requirements.'

The need for high-quality documentation and for contingency planning were motivated for students by the classroom equivalent of the real-world phenomenon of loss of staff members due to illness, death, relocation, etc. At the midpoint of the project, the student from each team judged, by the instructor, to be the team's strongest developer and another, randomly chosen, team member were removed from the team and reassigned to a different team. To evaluate the success of the staff change on students' approach to software engineering, after the case study project students were then asked to describe what they would have done differently had they understood that the real-world conditions under which they would be operating included the possibility of staff changes. About three quarters of the students mentioned the importance of up-to-date documentation, and about twenty percent had developed insight into appropriate utilization of staff, including the use of "understudies" and of preparation for the incorporation of new team members and thus demonstrated that they had learned the value of these processes.

Evaluation of how well the students internalized the need for solid requirements engineering was done at the end of the live-through case history. A written exam was based on another case history. This case history included a more difficult requirements engineering problem than that of the overdue book notice project. About three quarters of the students showed that they had mastered the notion of hidden requirements, and about one third showed that they had achieved reasonable competence in requirements engineering; about ten percent showed extremely keen insight into the problem.

Claims

The innovative process of live-through case histories is more effective than the traditional teaching of the Software Engineering course. In the past students were given lectures, homework and exams based on a well-respected Software Engineering text. Then they were asked to develop a project. When they approached the project they could not readily apply the techniques they learned. Once they understood the need for the processes they relearned them as they tried to apply them.

Directions

The authors are asking those teaching software engineering to use these case histories in their courses and report on the results. Materials are available at web site www.NJCSE.org along with a complete paper describing the live-through approach in detail. Please participate in gathering data to support or refute the claims in this paper. It is our intent to use the experience of instructors in several venues to make anecdotal conclusions more meaningful and perhaps statistically significant. We invite those who agree with us to join a consortium for the purpose of creating additional case histories and helping to refine the process.

Author Contact Information

Larry Bernstein, Director
Stevens Institute of Technology
Department of Computer Science
Castle Point on Hudson
Hoboken, New Jersey, 07030

Phone: 201-216-5442
E-Mail: lbernstein@ieee.org
URL: http://attila.stevens-tech.edu/~lbernste/

Dr. David Klappholz
Stevens Institute of Technology
Department of Computer Science
Castle Point on Hudson
Hoboken, New Jersey, 07030

Phone: 201-216-5509
E-Mail:d.klappholz@worldnet.att.net

URL: http://www.cs.stevens-tech.edu/People/Faculty/David_Klappholz.htm

About the Authors

Lawrence Bernstein is a recognized expert in software technology, network architecture, network management software, software project management, and technology conversion. He is teaching graduate courses on Computer Networks and undergraduate Software Engineering at Stevens Institute of Technology in Hoboken, NJ. He had a 35-year distinguished career at Bell Laboratories in managing large software projects and since retirement heads his own consulting firm. At Bell Labs he became a Chief Technical Officer of the Operations Systems Business Unit and an Executive Director. In parallel with these Bell Labs positions he was the Operations Systems Vice President of AT&T Network Systems from 1992-1996.

Mr. Bernstein holds eight software patents, has published one book and many articles on software engineering. The IEEE selected two of his articles for inclusion in best paper compendiums.

David Klappholz has twenty-seven years of experience teaching computer science and performing and supervising technology research sponsored by such organizations as NSF, DOE, IBM Research, and The New Jersey Commission on Science and Technology. His major research has been in: programming languages for parallel- and super-computers, and software tools for parallelizing sequential code. He is currently an Associate Professor of Computer Science at Stevens Institute of Technology. His current research interests include the design of novel methods for motivating and teaching Software Engineering.

STN Vol. 5, No. 2
In This Issue

Lon R. Dean, STN Editor
Data & Analysis Center for Software
P.O. Box 1400
Rome, NY 13442-1400

Phone: 800-214-7921
Fax: 315-334-4964
E-mail: news-editor@dacs.dtic.mil

Data & Analysis Center for Software
P.O. Box 1400
Rome, NY 13442-1400

Return Service Requested

PRSRT STD
U.S. Postage
P A I D
Permit #566
UTICA, NY